A Logic of Semantic Representations for Shallow Parsing

A Logic of Semantic Representations for Shallow Parsing

Alexander Koller
Saarland University
Saarbrücken, Germany
koller@mmci.uni-saarland.de

Alex Lascarides
University of Edinburgh
Edinburgh, UK
alex@inf.ed.ac.uk

Abstract

One way to construct semantic representations in a robust manner is to enhance shallow language processors with semantic components. Here, we provide a model theory for a semantic formalism that is designed for this, namely Robust Minimal Recursion Semantics (RMRS). We show that RMRS supports a notion of entailment that allows it to form the basis for comparing the semantic output of different parses of varying depth.

1 Introduction

Representing semantics as a logical form that supports automated inference and model construction is vital for deeper language engineering tasks, such as dialogue systems. Logical forms can be obtained from hand-crafted deep grammars (Butt et al., 1999; Copestake and Flickinger, 2000) but this lacks robustness: not all words and constructions are covered and by design ill-formed phrases fail to parse. There has thus been a trend recently towards robust wide-coverage semantic construction (e.g., (Bos et al., 2004; Zettlemoyer and Collins, 2007)). But there are certain semantic phenomena that these robust approaches don't capture reliably, including quantifier scope, optional arguments, and long-distance dependencies (for instance, Clark et al. (2004) report that the parser used by Bos et al. (2004) yields 63% accuracy on object extraction; e.g., *the man that I met...*). Forcing a robust parser to make a decision about these phenomena can therefore be error-prone. Depending on the application, it may be preferable to give the parser the option to leave a semantic decision open when it's not sufficiently informed—i.e., to compute a partial semantic representation and to complete it later, using information extraneous to the parser.

In this paper, we focus on an approach to semantic representation that supports this strategy: Robust Minimal Recursion Semantics (RMRS, Copestake (2007a)). RMRS is designed to support underspecification of lexical information, scope, and predicate-argument structure. It is an emerging standard for representing partial semantics, and has been applied in several implemented systems. For instance, Copestake (2003) and Frank (2004) use it to specify semantic components to shallow parsers ranging in depth from POS taggers to chunk parsers and intermediate parsers such as RASP (Briscoe et al., 2006). MRS analyses (Copestake et al., 2005) derived from deep grammars, such as the English Resource Grammar (ERG, (Copestake and Flickinger, 2000)) are special cases of RMRS. But RMRS, unlike MRS and related formalisms like dominance constraints (Egg et al., 2001), is able to express semantic information in the absence of full predicate argument structure and lexical subcategorisation.

The key contribution we make is to cast RMRS, for the first time, as a logic with a well-defined model theory. Previously, no such model theory existed, and so RMRS had to be used in a somewhat ad-hoc manner that left open exactly what any given RMRS representation actually *means*. This has hindered practical progress, both in terms of understanding the relationship of RMRS to other frameworks such as MRS and predicate logic and in terms of the development of efficient algorithms. As one application of our formalisation, we use entailment to propose a novel way of characterising consistency of RMRS analyses across different parsers.

Section 2 introduces RMRS informally and illustrates why it is necessary and useful for representing semantic information across deep and shallow language processors. Section 3 defines the syntax and model-theory of RMRS. We finish in Section 4 by pointing out some avenues for future research.

Proceedings of the 12th Conference of the European Chapter of the ACL, pages 451–459,
Athens, Greece, 30 March – 3 April 2009.

2 Deep and shallow semantic construction

Consider the following (toy) sentence:

(1) Every fat cat chased some dog.

It exhibits several kinds of ambiguity, including a quantifier scope ambiguity and lexical ambiguities—e.g., the nouns "cat" and "dog" have 8 and 7 WordNet senses respectively. Simplifying slightly by ignoring tense information, two of its readings are shown as logical forms below; these can be represented as trees as shown in Fig. 1.

(2) $\_\mathsf{every\_q\_1}(x, \_\mathsf{fat\_j\_1}(e', x) \wedge \_\mathsf{cat\_n\_1}(x),$
$\_\mathsf{some\_q\_1}(y, \_\mathsf{dog\_n\_1}(y),$
$\_\mathsf{chase\_v\_1}(e, x, y)))$

(3) $\_\mathsf{some\_q\_1}(y, \_\mathsf{dog\_n\_2}(y),$
$\_\mathsf{every\_q\_1}(x, \_\mathsf{fat\_j\_1}(e', x) \wedge \_\mathsf{cat\_n\_2}(x),$
$\_\mathsf{chase\_v\_1}(e, x, y)))$

Now imagine trying to extract semantic information from the output of a part-of-speech (POS) tagger by using the word lemmas as lexical predicate symbols. Such a semantic representation is highly partial. It will use predicate symbols such as $\_\mathrm{cat\_n}$, which might resolve to the predicate symbols $\_\mathsf{cat\_n\_1}$ or $\_\mathsf{cat\_n\_2}$ in the complete semantic representation. (Notice the different fonts for the ambiguous and unambiguous predicate symbols.) But most underspecification formalisms (e.g., MRS (Copestake et al., 2005) and CLLS (Egg et al., 2001)) are unable to represent semantic information that is as partial as what we get from a POS tagger because they cannot underspecify predicate-argument structure. RMRS (Copestake, 2007a) is designed to address this problem. In RMRS, the information we get from the POS tagger is as follows:

(4) $l_1 : a_1 : \_\mathrm{every\_q}(x_1),$
$l_{41} : a_{41} : \_\mathrm{fat\_j}(e'),$
$l_{42} : a_{42} : \_\mathrm{cat\_n}(x_3)$
$l_5 : a_5 : \_\mathrm{chase\_v}(e),$
$l_6 : a_6 : \_\mathrm{some\_q}(x_6),$
$l_9 : a_9 : \_\mathrm{dog\_n}(x_7)$

This RMRS expresses only that certain predications are present in the semantic representation—it doesn't say anything about semantic scope, about most arguments of the predicates (e.g., $\_\mathrm{chase\_v}(e)$ doesn't say who chases whom), or about the coindexation of variables ($\_\mathrm{every\_q}$ binds the variable x_1, whereas $\_\mathrm{cat\_n}$ speaks about x_3), and it maintains the lexical ambiguities. Technically, it consists of six *elementary predications* (EPs), one for each word lemma in the sentence; each of them is prefixed by a *label* and an *anchor*, which are essentially variables that refer to nodes in the trees in Fig. 1. We can say that the two trees *satisfy* this RMRS because it is possible to map the labels and anchors in (4) into nodes in each tree and variable names like x_1 and x_3 into variable names in the tree in such a way that the predications of the nodes that labels and anchors denote are consistent with those in the EPs of (4)—e.g., l_1 and a_1 can map to the root of the first tree in Fig. 1, x_1 to x, and the root label $\_\mathsf{every\_q\_1}$ is consistent with the EP predicate $\_\mathrm{every\_q}$.

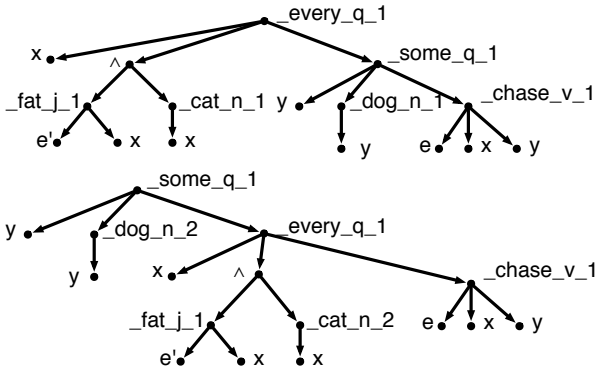

Figure 1: Semantic representations (2) and (3) as trees.

There are of course many other trees (and thus, fully specific semantic representations such as (2)) that are described equally well by the RMRS (4); this is not surprising, given that the semantic output from the POS tagger is so incomplete. If we have information about subjects and objects from a chunk parser like Cass (Abney, 1996), we can represent it in a more detailed RMRS:

(5) $l_1 : a_1 : \_\mathrm{every\_q}(x_1),$
$l_{41} : a_{41} : \_\mathrm{fat\_j}(e'),$
$l_{42} : a_{42} : \_\mathrm{cat\_n}(x_3)$
$l_5 : a_5 : \_\mathrm{chase\_v}(e),$
$\mathsf{ARG}_1(a_5, x_4), \mathsf{ARG}_2(a_5, x_5)$
$l_6 : a_6 : \_\mathrm{some\_q}(x_6),$
$l_9 : a_9 : \_\mathrm{dog\_n}(x_7)$
$x_3 = x_4, x_5 = x_7$

This introduces two new types of atoms. $x_3 = x_4$ means that x_3 and x_4 map to the same variable in any fully specific logical form; e.g., both to the variable x in Fig. 1. $\mathsf{ARG}_i(a, z)$ (and $\mathsf{ARG}_i(a, h)$)

express that the i-th child (counting from 0) of the node to which the anchor a refers is the variable name that z denotes (or the node that the *hole* h denotes). So unlike earlier underspecification formalisms, RMRS can specify the predicate of an atom separately from its arguments; this is necessary for supporting parsers where information about lexical subcategorisation is absent. If we also allow atoms of the form $\mathsf{ARG}_{\{2,3\}}(a, x)$ to express uncertainty as to whether x is the second or third child of the anchor a, then RMRS can even specify the arguments to a predicate while underspecifying their position. This is useful for specifying arguments to _give_v when a parser doesn't handle unbounded dependencies and is faced with *Which bone did you give the dog?* vs. *To which dog did you give the bone?*

Finally, the RMRS (6) is a notational variant of the MRS derived by the ERG, a wide-coverage deep grammar:

(6) $l_1 : a_1$:_every_q_1(x_1),
$\mathsf{RSTR}(a_1, h_2), \mathsf{BODY}(a_1, h_3)$
$l_{41} : a_{41}$:_fat_j_1(e'), $\mathsf{ARG}_1(a_{41}, x_2)$
$l_{42} : a_{42}$:_cat_n_1(x_3)
$l_5 : a_5$:_chase_v_1(e),
$\mathsf{ARG}_1(a_5, x_4), \mathsf{ARG}_2(a_5, x_5)$
$l_6 : a_6$:_some_q_1(x_6),
$\mathsf{RSTR}(a_6, h_7), \mathsf{BODY}(a_6, h_8)$
$l_9 : a_9$:_dog_n_1(x_7)
$h_2 =_q l_{42}, l_{41} = l_{42}, h_7 =_q l_9$
$x_1 = x_2, x_2 = x_3, x_3 = x_4$,
$x_5 = x_6, x_5 = x_7$

RSTR and BODY are conventional names for the ARG_1 and ARG_2 of a quantifier predicate symbol. Atoms like $h_2 =_q l_{42}$ ("qeq") specify a certain kind of "outscopes" relationship between the hole and the label, and are used here to underspecify the scope of the two quantifiers. Notice that the labels of the EPs for "fat" and "cat" are stipulated to be equal in (6), whereas the anchors are not. In the tree, it is the anchors that are mapped to the nodes with the labels _fat_j_1 and _cat_n_1; the label is mapped to the conjunction node just above them. In other words, the role of the anchor in an EP is to connect a predicate to its arguments, while the role of the label is to connect the EP to the surrounding formula. Representing conjunction with label sharing stems from MRS and provides compact representations.

Finally, (6) uses predicate symbols like _dog_n_1 that are meant to be more specific than symbols like _dog_n which the earlier RMRSs used. This reflects the fact that the deep grammar performs some lexical disambiguation that the chunker and POS tagger don't. The fact that the former symbol should be more specific than the latter can be represented using SPEC atoms like _dog_n_1 $\sqsubseteq$ _dog_n. Note that even a deep grammar will not fully disambiguate to semantic predicate symbols, such as WordNet senses, and so _dog_n_1 can still be consistent with multiple symbols like _dog_n_1 and _dog_n_2 in the semantic representation. However, unlike the output of a POS tagger, an RMRS symbol that's output by a deep grammar is consistent with symbols that all have the *same* arity, because a deep grammar fully determines lexical subcategorisation.

In summary, RMRS allows us to represent in a uniform way the (partial) semantics that can be extracted from a wide range of NLP tools. This is useful for hybrid systems which exploit shallower analyses when deeper parsing fails, or which try to match deeply parsed queries against shallow parses of large corpora; and in fact, RMRS is gaining popularity as a practical interchange format for exactly these purposes (Copestake, 2003). However, RMRS is still relatively ad-hoc in that its formal semantics is not defined; we don't know, formally, what an RMRS *means* in terms of semantic representations like (2) and (3), and this hinders our ability to design efficient algorithms for processing RMRS. The purpose of this paper is to lay the groundwork for fixing this problem.

3 Robust Minimal Recursion Semantics

We will now make the basic ideas from Section 2 precise. We will first define the syntax of the RMRS language; this is a notational variant of earlier definitions in the literature. We will then define a model theory for our version of RMRS, and conclude this section by carrying over the notion of *solved forms* from CLLS (Egg et al., 2001).

3.1 RMRS Syntax

We define RMRS syntax in the style of CLLS (Egg et al., 2001). We assume an infinite set of *node variables* $\mathsf{NVar} = \{X, Y, X_1, \ldots\}$, used as labels, anchors, and holes; the distinction between these will come from their position in the formulas. We also assume an infinite set of *base variables* BVar, consisting of individual variables $\{x, x_1, y, \ldots\}$ and event variables $\{e_1, \ldots\}$, and a vocabulary of

predicate symbols $\mathsf{Pred} = \{P, Q, P_1, \ldots\}$. RMRS formulas are defined as follows.

Definition 1. *An* RMRS *is a finite set φ of atoms of one of the following forms; $S \subseteq \mathbb{N}$ is a set of numbers that is either finite or $\mathbb{N}$ itself (throughout the paper, we assume $0 \in \mathbb{N}$).*

$$
\begin{array}{rcl}
A & ::= & X{:}Y{:}P \\
& | & \mathsf{ARG}_S(X, v) \\
& | & \mathsf{ARG}_S(X, Y) \\
& | & X \lhd^* Y \\
& | & v_1 = v_2 \mid v_1 \neq v_2 \\
& | & X = Y \mid X \neq Y \\
& | & P \sqsubseteq Q
\end{array}
$$

A node variable X is called a label *iff φ contains an atom of the form $X{:}Y{:}P$ or $Y \lhd^* X$; it is an* anchor *iff φ contains an atom of the form $Y{:}X{:}P$ or $\mathsf{ARG}_S(X, i)$; and it is a* hole *iff φ contains an atom of the form $\mathsf{ARG}_S(Y, X)$ or $X \lhd^* Y$.*

Def. 1 combines similarities to earlier presentations of RMRS (Copestake, 2003; Copestake, 2007b) and to CLLS/dominance constraints (Egg et al., 2001). For the most part, our syntax generalises that of older versions of RMRS: We use $\mathsf{ARG}_{\{i\}}$ (with a singleton set S) instead of ARG_i and $\mathsf{ARG}_{\mathbb{N}}$ instead of ARG_n, and the EP $l{:}a{:}P(v)$ (as in Section 2) is an abbreviation of $\{l{:}a{:}P, \mathsf{ARG}_{\{0\}}(a, v)\}$. Similarly, we don't assume that labels, anchors, and holes are syntactically different objects; they receive their function from their positions in the formula. One major difference is that we use dominance ($\lhd^*$) rather than qeq; see Section 3.4 for a discussion. Compared to dominance constraints, the primary difference is that we now have a mechanism for representing lexical ambiguity, and we can specify a predicate and its arguments separately.

3.2 Model Theory

The model theory formalises the relationship between an RMRS and the fully specific, alternative logical forms that it describes, expressed in the base language. We represent such a logical form as a tree τ, such as the ones in Fig. 1, and we can then define satisfaction of formulas in the usual way, by taking the tree as a model structure that interprets all predicate symbols specified above.

In this paper, we assume for simplicity that the base language is as in MRS; essentially, τ becomes the structure tree of a formula of predicate logic. We assume that Σ is a ranked signature consisting of the symbols of predicate logic: a unary constructor $\neg$ and binary constructors $\wedge$, $\rightarrow$, etc.; a set of 3-place quantifier symbols such as _every_q_1 and _some_q_1 (with the children being the bound variable, the restrictor, and the scope); and constructors of various arities for the predicate symbols; e.g., _chase_v_1 is of arity 3. Other base languages may require a different signature Σ and/or a different mapping between formulas and trees; the only strict requirement we make is that the signature contains a binary constructor $\wedge$ to represent conjunction. We write Σ_i and $\Sigma_{\geq i}$ for the set of all constructors in Σ with arity i and at least i, respectively. We will follow the typographical convention that non-logical symbols in Σ are written in sans-serif, as opposed to the RMRS predicate symbols like _cat_n and _cat_n_1.

The models of RMRS are then defined to be finite constructor trees (see also (Egg et al., 2001)):

Definition 2. *A* finite constructor tree *τ is a function $\tau : D \rightarrow \Sigma$ such that D is a* tree domain *(i.e., a subset of $\mathbb{N}^*$ which is closed under prefix and left sibling) and the number of children of each node $u \in D$ is equal to the arity of $\tau(u)$.*

We write $D(\tau)$ for the tree domain of a constructor tree τ, and further define the following relations between nodes in a finite constructor tree:

Definition 3. *$u \lhd^* v$* (dominance) *iff u is a prefix of v, i.e. the node u is equal to or above the node v in the tree. $u \lhd^*_\wedge v$ iff $u \lhd^* v$, and all symbols on the path from u to v (not including v) are $\wedge$.*

The *satisfaction* relation between an RMRS φ and a finite constructor tree τ is defined in terms of several assignment functions. First, a node variable assignment function $\alpha : \mathsf{NVar} \rightarrow D(\tau)$ maps the node variables in an RMRS to the nodes of τ. Second, a base language assignment function $g : \mathsf{BVar} \rightarrow \Sigma_0$ maps the base variables to nullary constructors representing variables in the base language. Finally, a function σ from Pred to the power set of $\Sigma_{\geq 1}$ maps each RMRS predicate symbol to a set of constructors from Σ. As we'll see shortly, this function allows an RMRS to underspecify lexical ambiguities.

Definition 4. *Satisfaction of atoms is defined as*

follows:

$\tau, \alpha, g, \sigma \models X{:}Y{:}P$ *iff*
$\quad \tau(\alpha(Y)) \in \sigma(P)$ *and* $\alpha(X) \lhd^*_\wedge \alpha(Y)$
$\tau, \alpha, g, \sigma \models \mathsf{ARG}_S(X, a)$ *iff exists* $i \in S$ *s.t.*
$\quad \alpha(X) \cdot i \in D(\tau)$ *and* $\tau(\alpha(X) \cdot i) = g(a)$
$\tau, \alpha, g, \sigma \models \mathsf{ARG}_S(X, Y)$ *iff exists* $i \in S$ *s.t.*
$\quad \alpha(X) \cdot i \in D(\tau), \alpha(X) \cdot i = \alpha(Y)$
$\tau, \alpha, g, \sigma \models X \lhd^* Y$ *iff* $\alpha(X) \lhd^* \alpha(Y)$
$\tau, \alpha, g, \sigma \models X =/\neq Y$ *iff* $\alpha(X) =/\neq \alpha(Y)$
$\tau, \alpha, g, \sigma \models v_1 =/\neq v_2$ *iff* $g(v_1) =/\neq g(v_2)$
$\tau, \alpha, g, \sigma \models P \sqsubseteq Q$ *iff* $\sigma(P) \subseteq \sigma(Q)$

A 4-tuple τ, α, g, σ *satisfies an* RMRS φ *(written* $\tau, \alpha, g, \sigma \models \varphi$*) iff it satisfies all of its elements.*

Notice that one RMRS may be satisfied by multiple trees; we can take the RMRS to be a partial description of each of these trees. In particular, RMRSs may represent semantic scope ambiguities and/or missing information about semantic dependencies, lexical subcategorisation and lexical senses. For $j = \{1, 2\}$, suppose that $\tau_j, \alpha_j, g_j, \sigma \models \varphi$. Then φ exhibits a semantic scope ambiguity if there are variables $Y, Y' \in \mathsf{NVar}$ such that $\alpha_1(Y) \lhd^* \alpha_1(Y')$ and $\alpha_2(Y') \lhd^* \alpha_2(Y)$. It exhibits missing information about semantic dependencies if there are base-language variables $v, v' \in \mathsf{BVar}$ such that $g_1(v) = g_1(v')$ and $g_2(v) \neq g_2(v')$. It exhibits missing lexical subcategorisation information if there is a $Y \in \mathsf{NVar}$ such that $\tau_1(\alpha_1(Y))$ is a constructor of a different type from $\tau_2(\alpha_2(Y))$ (i.e., the constructors are of a different arity or they differ in whether their arguments are scopal vs. nonscopal). And it exhibits missing lexical sense information if $\tau_1(\alpha_1(Y))$ and $\tau_2(\alpha_2(Y))$ are different base-language constructors, but of the same type.

Let's look again at the RMRS (4). This is satisfied by the trees in Fig. 1 (among others) together with some particular α, g, and σ. For instance, consider the left-hand side tree in Fig. 1. The RMRS (4) satisfies this tree with an assignment function α that maps the variables l_1 and a_1 to the root node, l_{41} and l_{42} to its second child (labeled with "$\wedge$"), a_{41} to the first child of *that* node (i.e. the node 21, labelled with "fat") and a_{42} to the node 22, and so forth. g will map x_1 and x_3 to x, and x_6 and x_7 to y, and so on. And σ will map each RMRS predicate symbol (which represents a word) to the set of its fully resolved meanings, e.g. _cat_n to a set containing _cat_n_1

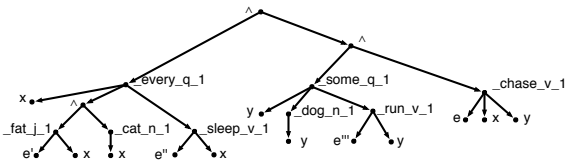

Figure 2: Another tree which satisfies (6).

and possibly others. It is then easy to verify that every single atom in the RMRS is satisfied—most interestingly, the EPs $l_{41}{:}a_{41}{:}\_\mathrm{fat\_j}(e')$ and $l_{42}{:}a_{42}{:}\_\mathrm{cat\_n}(x_3)$ are satisfied because $\alpha(l_{41}) \lhd^*_\wedge \alpha(a_{41})$ and $\alpha(l_{42}) \lhd^*_\wedge \alpha(a_{42})$.

Truth, validity and entailment can now be defined in terms of satisfiability in the usual way:

Definition 5. truth: $\tau \models \varphi$ *iff* $\exists \alpha, g, \sigma$ *such that* $\tau, \alpha, g, \sigma \models \varphi$

validity: $\models \varphi$ *iff* $\forall \tau$, $\tau \models \varphi$.

entailment: $\varphi \models \varphi'$ *iff* $\forall \tau$, *if* $\tau \models \varphi$ *then* $\tau \models \varphi'$.

3.3 Solved Forms

One aspect in which our definition of RMRS is like dominance constraints and unlike MRS is that any satisfiable RMRS has an infinite number of models which only differ in the areas that the RMRS didn't "talk about". Reading (6) as an MRS or as an RMRS of the previous literature, this formula is an instruction to build a semantic representation out of the pieces for "every fat cat", "some dog", and "chased"; a semantic representation as in Fig. 2 would *not* be taken as described by this RMRS. However, under the semantics we proposed above, this tree is a correct model of (6) because all atoms are still satisfied; the RMRS didn't say anything about "sleep" or "run", but it couldn't enforce that the tree shouldn't contain those subformulas either.

In the context of robust semantic processing, this is a desirable feature, because it means that when we enrich an RMRS obtained from a shallow processor with more semantic information—such as the relation symbols introduced by syntactic constructions such as appositives, noun-noun compounds and free adjuncts—we don't change the set of models; we only *restrict* the set of models further and further towards the semantic representation we are trying to reconstruct. Furthermore, it has been shown in the literature that a dominance-constraint style semantics for underspecified representations gives us more room to

manoeuvre when developing efficient solvers than an MRS-style semantics (Althaus et al., 2003).

However, enumerating an infinite number of models is of course infeasible. For this reason, we will now transfer the concept of *solved forms* from dominance constraints to RMRS. An RMRS in solved form is guaranteed to be satisfiable, and thus each solved form represents an infinite class of models. However, each satisfiable RMRS has only a finite number of solved forms which partition the space of possible models into classes such that models within a class differ only in 'irrelevant' details. A solver can then enumerate the solved forms rather than all models.

Intuitively, an RMRS in solved form is fully specified with respect to the predicate-argument structure, all variable equalities and inequalities and scope ambiguities have been resolved, and only lexical sense ambiguities remain. This is made precise below.

Definition 6. *An RMRS φ is* in solved form *iff:*

1. *every variable in φ is either a hole, a label or an anchor (but not two of these);*
2. *φ doesn't contain equality, inequality, and SPEC ($\sqsubseteq$) atoms;*
3. *if $\mathsf{ARG}_S(Y, i)$ is in φ, then $|S| = 1$;*
4. *for any label Y and index set S, there are no two atoms $\mathsf{ARG}_S(Y, i)$ and $\mathsf{ARG}_S(Y, i')$ in φ;*
5. *if Y is an anchor in some EP $X{:}Y{:}P$ and k is the maximum number such that $\mathsf{ARG}_{\{k\}}(X, i)$ is in φ for any i, then there is a constructor $p \in \sigma(P)$ whose arity is at least k;*
6. *no label occurs on the right-hand side of two different $\lhd^*$ atoms.*

Because solved forms are so restricted, we can 'read off' at least one model from each solved form:

Proposition 1. *Every* RMRS *in solved form is satisfiable.*

Proof (sketch; see also (Duchier and Niehren, 2000)). For each EP, we choose to label the anchor with the constructor p of sufficiently high arity whose existence we assumed; we determine the edges between an anchor and its children from the uniquely determined ARG atoms; plugging labels into holes is straightforward because no label is dominated by more than one hole; and spaces between the labels and anchors are filled with conjunctions. □

We can now define the solved forms *of* an RMRS φ; these finitely many RMRSs in solved form partition the space of models of φ into classes of models with trivial differences.

Definition 7. *The* syntactic dominance relation $D(\varphi)$ *in an* RMRS φ *is the reflexive, transitive closure of the binary relation*

$$\{(X, Y) \quad | \quad \varphi \text{ contains } X \lhd^* Y \text{ or } ARG_S(X, Y) \text{ for some } S\}$$

An RMRS φ' is a solved form of *the RMRS φ iff φ' is in solved form and there is a substitution s that maps the node and base variables of φ to the node and base variables of φ' such that*

1. *φ' contains the* EP $X'{:}Y'{:}P$ *iff there are variables X, Y such that $X{:}Y{:}P$ is in φ, $X' = s(X)$, and $Y' = s(Y)$;*
2. *for every atom $\mathsf{ARG}_S(X, i)$ in φ, there is exactly one atom $\mathsf{ARG}_{S'}(X', i')$ in φ' with $X' = s(X)$, $i' = s(i)$, and $S' \subseteq S$;*
3. *$D(\varphi') \supseteq s(D(\varphi))$.*

Proposition 2. *For every tuple $(\tau, \alpha, g, \sigma)$ that satisfies some* RMRS *φ, there is a solved form φ' of φ such that $(\tau, \alpha, g, \sigma)$ also satisfies φ'.*

Proof. We construct the substitution s from α and g. Then we add all dominance atoms that are satisfied by α and restrict the ARG atoms to those child indices that are actually used in τ. The result is in solved form because τ is a tree; it is a solved form of φ by construction. □

Proposition 3. *Every* RMRS *φ has only a finite number of solved forms, up to renaming of variables.*

Proof. Up to renaming of variables, there is only a finite number of substitutions on the node and base variables of φ. Let s be such a substitution. This fixes the set of EPs of any solved form of φ that is based on s uniquely. There is only a finite set of choices for the subsets S' in condition 2 of Def. 7, and there is only a finite set of choices of new dominance atoms that satisfy condition 3. Therefore, the set of solved forms of φ is finite. □

Let's look at an example for all these definitions. All the RMRSs presented in Section 2 (replacing $=_q$ by $\lhd^*$) are in solved form; this is least obvious for (6), but becomes clear once we notice that no label is on the right-hand side of two dominance atoms. However, the model constructed in the proof of Prop. 1 looks a bit like Fig. 2; both models are problematic in several ways and in particular contain an unbound variable y even though they also contains a quantifier that binds y. If we restrict the class of models to those in which such variables are bound (as Copestake et al. (2005) do), we can enforce that the quantifiers outscope their bound variables without changing models of the RMRS further—i.e., we add the atoms $h_3 \lhd^* l_5$ and $h_8 \lhd^* l_5$. Fig. 2 is no longer a model for the extended RMRS, which in turn is no longer in solved form because the label l_5 is on the right-hand side of two dominance atoms. Instead, it has the following two solved forms:

(7) $l_1{:}a_1{:}\_\text{every\_q\_1}(x_1)$,
$\mathsf{RSTR}(a_1, h_2)$, $\mathsf{BODY}(a_1, h_3)$,
$l_{41}{:}a_{41}{:}\_\text{fat\_j\_1}(e')$, $\mathsf{ARG}_1(a_{41}, x_1)$,
$l_{41}{:}a_{42}{:}\_\text{cat\_n\_1}(x_1)$,
$l_6{:}a_6{:}\_\text{some\_q\_1}(x_6)$,
$\mathsf{RSTR}(a_6, h_7)$, $\mathsf{BODY}(a_6, h_8)$,
$l_9{:}a_9{:}\_\text{dog\_n\_1}(x_6)$,
$l_5{:}a_5{:}\_\text{chase\_v\_1}(e)$,
$\mathsf{ARG}_1(a_5, x_1)$, $\mathsf{ARG}_2(a_5, x_6)$,
$h_2 \lhd^* l_{41}$, $h_3 \lhd^* l_6$, $h_7 \lhd^* l_9$, $h_8 \lhd^* l_5$

(8) $l_1{:}a_1{:}\_\text{every\_q\_1}(x_1)$,
$\mathsf{RSTR}(a_1, h_2)$, $\mathsf{BODY}(a_1, h_3)$,
$l_{41}{:}a_{41}{:}\_\text{fat\_j\_1}(e')$, $\mathsf{ARG}_1(a_{41}, x_1)$,
$l_{41}{:}a_{42}{:}\_\text{cat\_n\_1}(x_1)$,
$l_6{:}a_6{:}\_\text{some\_q\_1}(x_6)$,
$\mathsf{RSTR}(a_6, h_7)$, $\mathsf{BODY}(a_6, h_8)$,
$l_9{:}a_9{:}\_\text{dog\_n\_1}(x_6)$,
$l_5{:}a_5{:}\_\text{chase\_v\_1}(e)$,
$\mathsf{ARG}_1(a_5, x_1)$, $\mathsf{ARG}_2(a_5, x_6)$,
$h_2 \lhd^* l_{41}$, $h_3 \lhd^* l_5$, $h_7 \lhd^* l_9$, $h_8 \lhd^* l_1$

Notice that we have eliminated all equalities by unifying the variable names, and we have fixed the relative scope of the two quantifiers. Each of these solved forms now stands for a separate class of models; for instance, the first model in Fig. 1 is a model of (7), whereas the second is a model of (8).

3.4 Extensions

So far we have based the syntax and semantics of RMRS on the dominance relation from Egg et al. (2001) rather than the qeq relation from Copestake et al. (2005). This is partly because dominance is the weaker relation: If a dependency parser links a determiner to a noun and this noun to a verb, then we can use dominance but not qeq to represent that the predicate introduced by the verb is outscoped by the quantifier introduced by the determiner (see earlier discussion). However, it is very straightforward to extend the syntax and semantics of the language to include the qeq relation. This extension adds a new atom $X =_q Y$ to Def. 1, and τ, α, g, σ will satisfy $X =_q Y$ iff $\alpha(X) \lhd^* \alpha(Y)$, each node on the path is a quantifier, and each step in the path goes to the rightmost child. All the above propositions about solved forms still hold if "dominance" is replaced with "qeq".

Furthermore, grammar developers such as those in the DELPH-IN community typically adopt conventions that restrict them to a fragment of the language from Def. 1 (once qeq is added to it), or they restrict attention to only a subset of the models (e.g., ones with correctly bound variables, or ones which don't contain extra material like Fig. 2). Our formalism provides a general framework into which all these various fragments fit, and it's a matter of future work to explore these fragments further.

Another feature of the existing RMRS literature is that each term of an RMRS is equipped with a *sort*. In particular, individual variables x, event variables e and holes h are arranged together with their subsorts (e.g., e_{past}) and supersorts (e.g., sort i abstracts over x and e) into a sort hierarchy $\mathcal{S}$. For simplicity we defined RMRS without sorts, but it is straightforward to add them. For this, one assumes that the signature Σ is sorted, i.e. assigns a sort $s_1 \times \ldots s_n \to s$ to each constructor, where n is the constructor's arity (possibly zero) and $s, s_1, \ldots, s_n \in \mathcal{S}$ are atomic sorts. We restrict the models of RMRS to trees that are *well-sorted* in the usual sense, i.e. those in which we can infer a sort for each subtree, and require that the variable assignment functions likewise respect the sorts. If we then modify Def. 6 such that the constructor p of sufficiently high arity is also consistent with the sorts of the known arguments—i.e., if p has sort $s_1 \times \ldots \times s_n \to s$ and the RMRS contains an atom $\mathsf{ARG}_{\{k\}}(Y, i)$ and i is of sort s', then s' is a subsort of s_k—all the above propositions about solved forms remain true.

4 Future work

The above definitions serve an important theoretical purpose: they formally underpin the use of RMRS in practical systems. Next to the peace of mind that comes with the use of a well-understood formalism, we hope that the work reported here will serve as a starting point for future research.

One direction to pursue from this paper is the development of efficient solvers for RMRS. As a first step, it would be interesting to define a practically useful fragment of RMRS with polynomial-time satisfiability. Our definition is sufficiently close to that of dominance constraints that we expect that it should be feasible to carry over the definition of *normal dominance constraints* (Althaus et al., 2003) to RMRS; neither the lexical ambiguity of the node labels nor the separate specification of predicates and arguments should make satisfiability harder.

Furthermore, the above definition of RMRS provides new concepts which can help us phrase questions of practical grammar engineering in well-defined formal terms. For instance, one crucial issue in developing a hybrid system that combines or compares the outputs of deep and shallow processors is to determine whether the RMRSs produced by the two systems are compatible. In the new formal terms, we can characterise compatibility of a more detailed RMRS φ (perhaps from a deep grammar) and a less detailed RMRS φ' simply as *entailment* $\varphi \models \varphi'$. If entailment holds, this tells us that all claims that φ' makes about the semantic content of a sentence are consistent with the claims that φ makes.

At this point, we cannot provide an efficient algorithm for testing entailment of RMRS. However, we propose the following novel syntactic characterisation as a starting point for research along those lines. We call an RMRS φ' an *extension* of the RMRS φ if φ' contains all the EPs of φ and $D(\varphi') \supseteq D(\varphi)$.

Proposition 4. *Let φ, φ' be two* RMRS*s. Then $\varphi \models \varphi'$ iff for every solved form S of φ, there is a solved form S' of φ' such that S is an extension of S'.*

Proof (sketch). "$\Leftarrow$" follows from Props. 1 and 2.

"$\Rightarrow$": We construct a solved form for φ' by choosing a solved form for φ and appropriate substitutions for mapping the variables of φ and φ' onto each other, and removing all atoms using variables that don't occur in φ' . The hard part is the proof that the result is a solved form of φ'; this step involves proving that if $\varphi \models \varphi'$ with the same variable assignments, then all EPs in φ' also occur in φ. □

5 Conclusion

In this paper, we motivated and defined RMRS—a semantic framework that has been used to represent, compare, and combine semantic information computed from deep and shallow parsers. RMRS is designed to be maximally flexible on the type of semantic information that can be left underspecified, so that the semantic output of a shallow parser needn't over-determine or under-determine the semantics that can be extracted from the shallow syntactic analysis. Our key contribution was to lay the formal foundations for a formalism that is emerging as a standard in robust semantic processing.

Although we have not directly provided new tools for modelling or processing language, we believe that a cleanly defined model theory for RMRS is a crucial prerequisite for the future development of such tools; this strategy was highly successful for dominance constraints (Althaus et al., 2003). We hope that future research will build upon this paper to develop efficient algorithms and implementations for solving RMRSs, performing inferences that enrich RMRSs from shallow analyses with deeper information, and checking consistency of RMRSs that were obtained from different parsers.

Acknowledgments. We thank Ann Copestake, Dan Flickinger, and Stefan Thater for extremely fruitful discussions and the reviewers for their comments. The work of Alexander Koller was funded by a DFG Research Fellowship and the Cluster of Excellence "Multimodal Computing and Interaction".

References

S. Abney. 1996. Partial parsing via finite-state cascades. In John Carroll, editor, *Workshop on Robust Parsing (ESSLLI-96)*, pages 8–15, Prague.

E. Althaus, D. Duchier, A. Koller, K. Mehlhorn, J. Niehren, and S. Thiel. 2003. An efficient graph algorithm for dominance constraints. *J. Algorithms*, 48:194–219.

J. Bos, S. Clark, M. Steedman, J. Curran, and J. Hockenmaier. 2004. Wide coverage semantic representations from a CCG parser. In *Proceedings of the International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland.

E.J. Briscoe, J. Carroll, and R. Watson. 2006. The second release of the rasp system. In *Proceedings of the COLING/ACL 2006 Interaction Presentation Sessions*, Sydney, Australia.

M. Butt, T. Holloway King, M. Niño, and F. Segond. 1999. *A Grammar Writer's Cookbook*. CSLI Publications.

S. Clark, M. Steedman, and J. Curran. 2004. Object extraction and question parsing using CCG. In *Proceedings from the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 111–118, Barcelona.

A. Copestake and D. Flickinger. 2000. An open-source grammar development environment and english grammar using HPSG. In *Proceedings of the Second Conference on Language Resources and Evaluation (*LREC 2000*)*, pages 591–600, Athens.

A. Copestake, D. Flickinger, I. Sag, and C. Pollard. 2005. Minimal recursion semantics: An introduction. *Research on Language and Computation*, 3(2–3):281–332.

A. Copestake. 2003. Report on the design of RMRS. Technical Report EU Deliverable for Project number IST-2001-37836, WP1a, Computer Laboratory, University of Cambridge.

A. Copestake. 2007a. Applying robust semantics. In *Proceedings of the 10th Conference of the Pacific Assocation for Computational Linguistics (PACLING)*, pages 1–12, Melbourne. Invited talk.

A. Copestake. 2007b. Semantic composition with (robust) minimal recursion semantics. In *ACL-07 workshop on Deep Linguistic Processing*, pages 73–80, Prague.

D. Duchier and J. Niehren. 2000. Dominance constraints with set operators. In *In Proceedings of the First International Conference on Computational Logic (CL2000), LNCS*, pages 326–341. Springer.

M. Egg, A. Koller, and J. Niehren. 2001. The constraint language for lambda structures. *Journal of Logic, Language, and Information*, 10:457–485.

A. Frank. 2004. Constraint-based RMRS construction from shallow grammars. In *Proceedings of the International Conference in Computational Linguistics (COLING 2004)*, Geneva, Switzerland.

L. Zettlemoyer and M. Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687.